

Clustering Documents by Graph based AHC Variants

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—The AHC variants are modified into the graph-based versions as the text clustering algorithms. The initial AHC algorithm which clusters graphs, instead of numerical vectors, was previously proposed. In this research, in the case of the numerical vector-based versions, we mention the three AHC variants: one where the data clustering proceeds in the bottom-up direction with the similarity threshold, one where it allows any merge of more than two pairs, and one where clusters are merged, depending on the radius. In this research, we modify the three AHC variants into the graph-based versions as the approaches to the text clustering, as well as the initial AHC algorithm. The goal of this research is to improve the clustering performance by proposing the modified versions of the AHC variants.

I. INTRODUCTION

The text clustering refers to the process of segmenting a single group of texts into more than one subgroup which is called cluster, depending on their contents. In this research, it is assumed that all data items are initially given as a group and clustered at a time. The process of text clustering is to encode texts into structured data as text representations, define a similarity metric between text representations, and apply the unsupervised learning algorithm to the task. The scope of this research is set to the hard clustering where each text is allowed to belong to only one cluster, and this task will be expanded into the soft text clustering and the hierarchical text clustering in subsequent research. In future, the task relates to the word clustering for improving their clustering performances by their synergy effect.

This research is motivated by the upgraded performance of the graph based AHC algorithm in using it for the text clustering. The main problems in encoding texts into the numerical vectors are solved by encoding texts into graphs and defining the similarity metric between graphs. The AHC algorithm was modified into the graph-based version, using the similarity metric. The performance of the graph-based version was empirically validated as the better approach than the traditional version to the text clustering. The distinguishable points of this research are the modification of the AHC variants into the graph-based versions and the application of them to the text clustering.

The idea of this research is to modify the AHC variants into the graph-based versions as the approaches to the text clustering. The similarity metric between graphs which was

used for modifying the AHC algorithm in the previous work is adopted for modifying the AHC variants. The three AHC variants are one where data items are clustered depending on the parameter, similarity threshold, one where it is possible to allow merging more than one pair in each iteration, and one where clusters are merged depending on the radius from a particular cluster. The three AHC variants are modified into the graph-based versions and used for the text clustering. The AHC variant which integrates the three versions will be considered in subsequent research.

Let us mention some benefits from this research. Converting texts into graphs becomes the solution to the problems from encoding texts into numerical vectors. Replacing the AHC algorithm by the AHC variants is intended for upgrading the clustering speed. Modifying the AHC variants into their graph-based versions is intended for upgrading the clustering performance. This research is intended to provide the more recommendable approaches for implementing the text clustering system by improving both measures.

Let us mention the organization of this paper. In Section II, we explore the previous works which are relevant to this study. In Section III, we describe in detail what is proposed in this research. In Section IV, we mention the significances of this research and the remaining tasks. This paper is composed of the four sections.

II. PREVIOUS WORKS

This section is concerned with the previous works which are relevant to this research. It is intended to expand the style of modifying the AHC algorithm to its three variants. The directions of exploring previous works are derivation of AHC variants, modification of the standard AHC algorithm into its graph-based version, and the cases of encoding a text into a graph. The distinguishable points of this research are derivation of three AHC variants from the standard version, modification of them into graph-based versions, and their applications to the text clustering. This section is intended to explore previous works for providing the background for this research.

Let us explore the previous cases of deriving AHC variants. In 2012, Murtagh and Contreras mentioned the possibility of deriving various versions of AHC algorithm and a particular variant which uses nearest neighbors [3].

In 2013, Sasirekha and Baby presented various similarity metrics between vectors and strategies of defining a similarity between clusters [4]. In 2015, Nazari et al. proposed the AHC algorithm which merges two clusters based on their common nearest neighbors [5]. In the AHC algorithms which are mentioned in above previous literatures, only one pair of clusters is merged in each iteration.

Let us explore the previous works which are concerned with the application of the modified version of the standard AHC algorithm to text clustering. In 2017, Jo initially proposed that the graph-based version should be applied to the text clustering [7]. In 2019, Jo validated empirically its performances in the text clustering [8]. In 2023, he prepares the journal article which describes the graph-based AHC algorithm and its application to the text clustering for its publication [10]. What is provided by the previous works becomes the basis for this research.

Let us explore the previous works on representation of a text into a graph. In 2004, the process of representing a text into a graph was mentioned by Hensman [1]. In 2007, encoding a text into a graph for doing data mining tasks was covered by Jin and Srihari [2]. In 2020, the matching criteria of graphs which represent texts was proposed by Osman and Barukub [9]. The scheme of encoding a text into a graph and the similarity matrix between graphs may be different in this research.

Let us mention the differences of this research from the previous works which are explored above. In this research, we derive the AHC variants as fast clustering algorithms by allowing merging multiple cluster pairs for each iteration. The three AHC variants are modified into graph-based versions and adopted for implementing the text clustering system. In this research, the graph based AHC variants are developed as clustering algorithms as well as classification algorithms. We will describe the modified AHC variants in Section III.

III. PROPOSED WORKS

This section is concerned with what is proposed in this research. In Section III-A, we describe the process of encoding a text into a graph and the proposed similarity metric. In Section III-B, we describe the standard version of AHC algorithm where the proposed similarity metric is adopted. In Section III-C, we derive the three variants from the standard version. In Section III-D, we present the system architecture of the text clustering system.

A. Encoding Proccerss

This section is concerned with the graph as a text representation and the similarity between graphs. In representing a text into a graph, a word is given as a vertex, and a similarity between words is given as an edge. A graph is viewed as a set of edges, and the similarity between edges is computed before computing the similarity between graphs.

The similarity between them is computed by averaging the similarities of all possible edge pairs. This section is intended to describe the process of encoding a text into a graph and the process of computing the similarity between graphs.

The process of encoding a text into a graph is illustrated in Figure 1. Words are retrieved as the vertices from the text by indexing it. The similarities among words are computed as edges in the edge definition. In the edge selection, the edges with their higher similarities are selected for representing a text into a graph. Refer to [6] for studying the encoding process in more detail.

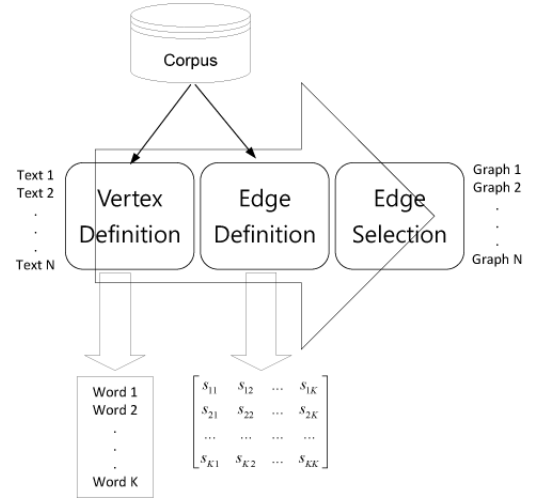


Figure 1. Process of Encoding Texts into Graphs

The three cases of computing the similarity between two edges are illustrated in Figure 2. Each weight which is associated with its own edge is assumed to be a normalized value between zero and one, and if both nodes are identical to each other, the average over two weights is the similarity between two edges. If either of the two nodes is identical to each other, the product of two weights is the similarity between two edges. If neither of two nodes is identical, zero is the similarity between them. The similarity between two edges is always a normalized value between zero and one.

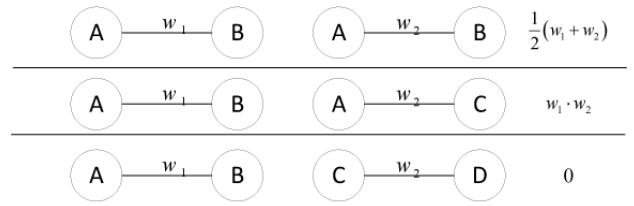


Figure 2. Three Cases of Comparing Two Edges with Each Other

Let us mention the process of computing the similarity between graphs as a semantic similarity between texts. The two graphs which represent texts are notated by edge sets, $G_1 =$

$\{e_{11}, e_{12}, \dots, e_{1|G_1|}\}$ and $G_2 = \{e_{21}, e_{22}, \dots, e_{2|G_2|}\}$. Two edges, $e_{1i} \in G_1$ and $e_{2j} \in G_2$, are associated with their own weights, w_{1i} and w_{2j} , and the similarity between two edges, e_{1i} and e_{2j} by equation (1), (2), or (3), depending on the cases which are mentioned above,

$$\text{sim}(e_{1i}, e_{2j}) = \frac{1}{2}(w_{1i} + w_{2j}) \quad (1)$$

$$\text{sim}(e_{1i}, e_{2j}) = w_{1i} \times w_{2j} \quad (2)$$

$$\text{sim}(e_{1i}, e_{2j}) = 0 \quad (3)$$

The similarity between two graphs is computed by equation (4),

$$\text{sim}(G_1, G_2) = \frac{1}{|G_1| \times |G_2|} \sum_{i=1}^{|G_1|} \sum_{j=1}^{|G_2|} \text{sim}(e_{1i}, e_{2j}). \quad (4)$$

The value which is generated from equation (4), is always given as a normalized value between zero and one as shown in equation (5),

$$0 \leq \text{sim}(G_1, G_2) \leq 1. \quad (5)$$

Let us make some remarks on the encoding process and the computation of the similarity between texts. A text is encoded into a graph by the vertex definition, the edge definition, and the edge weighting. The three cases are considered for computing the similarity between edges. The similarity between graphs is computed by equation (4). In the future research, we consider representing a text into multiple graphs.

B. Initial Version of Graph based AHC Algorithm

This section is concerned with the initial version of graph based AHC algorithm. The version of AHC algorithm was initially proposed as the approach to the text clustering by Jo in 2019 [8]. The similarity metric between graphs which was described in the previous section is used for computing the similarity between clusters. The AHC algorithm proceeds clustering data items with the bottom-up direction. This section is intended to describe the initial version of AHC algorithm from which its variants are derived.

The initial status for applying the AHC algorithm to the word clustering is illustrated in Figure 3. The words as clustering targets are encoded into graphs by the process, which was described in Section III-A, and they are notated by a set, $Tr = \{G_1, G_2, \dots, G_N\}$. The clusters are defined as many as data items, as $C_1, C_2, \dots, C_N$, and each cluster includes a data item as $C_i = \{G_i\}$. The initial status which is represented in Figure 00 is notated by $C_1 = \{G_1\}, C_2 = \{G_2\}, \dots, C_N = \{G_N\}$. The cluster with only one item is called singleton, and singletons are constructed as many as data items in the initial status.

The process of computing the similarity between two clusters is illustrated in Figure 4. The two clusters are notated by two sets of items, $C_1 = \{G_{11}, G_{12}, \dots, G_{1|C_1|}\}$



Figure 3. Initial Clusters in using AHC Algorithm: Singletons

and $C_2 = \{G_{21}, G_{22}, \dots, G_{2|C_2|}\}$. The similarity between clusters, C_1 and C_2 , is computed by equation (6),

$$\text{sim}(C_1, C_2) = \frac{1}{|C_1| \cdot |C_2|} \sum_{i=1}^{|C_1|} \sum_{j=1}^{|C_2|} \text{sim}(G_{1i}, G_{2j}). \quad (6)$$

If the similarity metric which was described in Section III-A is adopted, the similarity between two clusters is always a normalized value between zero and one, as expressed in equation (7),

$$0 \leq \text{sim}(C_1, C_2) \leq 1. \quad (7)$$

The complexity of computing the similarity between two clusters is expressed by $O(|C_1||C_2|)$.

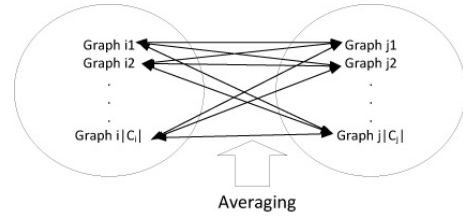


Figure 4. Similarity between Clusters

The process of clustering data items is illustrated as pseudo codes in Figure 5. The AHC algorithm begins with the status where singletons are given as many as data items. All possible pairs of clusters are generated from the current cluster list, their similarities are computed by equation (6), and the cluster pair with the highest similarity is merged into a cluster. In the AHC algorithm, the data items are clustered by iterating computing similarities of all possible cluster pairs and merge a cluster pair into a cluster. In each iteration, one cluster is decreased, and this iteration continues until reaching the desirable number of clusters.

Let us make some remarks on the initial version of the AHC algorithm from which we derive the variants. The initial status in applying the AHC algorithm to data clustering is that there are singletons as many as data items. The similarity between clusters is computed by averaging the similarities of all possible pairs from them. Data items are clustered by iterating computing the similarities of all possible cluster pairs and merge the pair with its maximal similarity into one cluster. In its variant, we will consider allow more than one pair of clusters to be merged.

C. AHC Variants

This section is concerned with the variants which are derived from the AHC algorithm which was covered in

```

List clusterDataItem(List graphList, int desiredClusterListSize){
    int itemSize = graphList.size();

    if(itemSize <= desiredClusterNumber)
        return null;
    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        Graph dataItem = graphList.getElement(i);
        cc.addDataItem(dataItem);
        clusterList.addElement(cc);
    }

    int clusterListSize = clusterList.size();

    //Cluster Data Items in the bottom up direction
    while(clusterList.size() > desiredClusterListSize){
        double maxSimilarity = 0.0;
        int maxIndex1 = 0;
        int maxIndex2 = 0;
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            for(int j = 0; j < clusterListSize; j++){
                Cluster c2 = clusterList.getElement(j);
                double similarity = sim(c1, c2);
                if(similarity > maxSimilarity){
                    maxSimilarity = similarity;
                    maxIndex1 = i;
                    maxIndex2 = j;
                }
            }
        }
        Cluster cmax1 = clusterList.getElement(maxIndex1);
        Cluster cmax2 = clusterList.getElement(maxIndex2);
        Cluster mergeCluster = cmax1.merge(cmax2);
        clusterList.deleteElement(cmax1);
        clusterList.deleteElement(cmax2);
        clusterList.addElement(mergeCluster);
    }
}

```

Figure 5. Initial Version of AHC Algorithm

Section III-B. The difference from the traditional version of AHC algorithm is to adopt the similarity metric between graphs for computing the similarity between clusters. In this section, we derive the three variants by merging cluster pairs by adopt the threshold-based selection, allowing merge of multiple pairs in each iteration, and merging clusters within the radius. In this research, we adopt the three variants of AHC algorithm for implementing the word clustering system. This section is intended to describe the three variants of AHC algorithm.

In Figure 6, we illustrate the process of clustering data items by the first variant of AHC algorithm. The clusters are initialized with singletons like the initial version which was described in Section III-B. All possible cluster pairs are generated, and the cluster pairs whose similarities are more than or equality to the similarity threshold are merged. If there is no pair which satisfies the condition, clustering data items is terminated. In this variant, the number of clusters is decreased variably in this variant.

We illustrate the process of clustering data items by the second variant of AHC algorithm in Figure 7 as a pseudo code. The number of cluster pairs which are merged into each iteration is given as an additional parameter in this variant. All possible cluster pairs are generated, and the similarity is computed for each pair. The cluster pairs are sorted by the descending order of the similarity, and the pairs within the rank are merged. The difference of this variant from the initial version is to merge multiple cluster pairs,

```

List clusterDataItem(List graphList, double similarityThreshold){
    int itemSize = graphList.size();

    if(itemSize <= desiredClusterNumber)
        return null;
    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        Graph dataItem = graphList.getElement(i);
        cc.addDataItem(dataItem);
        clusterList.addElement(cc);
    }

    boolean similarityFlag = false;

    //Cluster Data Items in the bottom up direction
    while(similarityFlag){
        int clusterListSize = clusterList.size();
        similarityFlag = false;
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            for(int j = 0; j < clusterListSize; j++){
                Cluster c2 = clusterList.getElement(j);
                double similarity = sim(c1, c2);
                if(similarity >= similarityThreshold){
                    Cluster mergeCluster = c1.merge(cmax2);
                    clusterList.updateElement(i, mergeCluster);
                    clusterList.deleteElement(j);
                    clusterListSize = clusterList.size();
                    similarityFlag = true;
                }
            }
        }
    }
}

```

Figure 6. Feature Similarity based AHC Variant 1: Similarity Threshold Based AHC Algorithm

instead of one pair.

```

List clusterDataItem(List graphList, double similarityThreshold){
    int itemSize = graphList.size();

    if(itemSize <= desiredClusterNumber)
        return null;
    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        Graph dataItem = graphList.getElement(i);
        cc.addDataItem(dataItem);
        clusterList.addElement(cc);
    }

    int clusterListSize = clusterList.size();

    //Cluster Data Items in the bottom up direction
    while(clusterListSize > desiredClusterSize){
        List pairList = new List();
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            for(int j = i; j < clusterListSize; j++){
                Cluster c2 = clusterList.getElement(j);
                Pair indivPair = new Pair(c1, c2);
                indivPair.computeSimilarity();
                pairList.addElement(indivPair);
            }
        }
        pairList.sortPairs();
        List selectedPairList = pairList.getSubList(0, rank-1);
        clusterList.mergeClusters(selectedPairList);
        clusterListSize = clusterList.size();
    }
}

```

Figure 7. Feature Similarity based AHC Variant 2: Merge of Multiple Pairs

In Figure 8, we illustrate the process of clustering data items by the last variant of AHC algorithm. The similarity threshold is given as an external parameter, and the number of other clusters whose similarity is greater than or equality to the similarity threshold is counted. In each iteration,

the cluster with its maximal number of its similar clusters is appointed as the pivot, and the pivot cluster and its similar ones are merged into one cluster. This variant iterates selecting the pivot cluster in the current cluster list and merge the pivot and its similar ones, as the process of clustering data items. If there is no cluster with its similar cluster, the iteration is terminated.

```

List clusterDataItem(List graphList, double similarityThreshold){
    int itemSize = graphList.size();

    if(itemSize <= desiredClusterNumber)
        return null;
    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        Graph dataItem = graphList.getElement(i);
        cc.addDataItem(dataItem);
        clusterList.addElement(cc);
    }

    //Cluster Data Items in the bottom up direction
    while(true){
        int clusterListSize = clusterList.size();
        if(clusterListSize == 1)
            return;
        int maxSimilarClusterSize = 0;
        int maxIndex = 0;
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            int similarClusterSize = c1.countSimilarClusters(clusterList);
            if(similarClusterSize > maxSimilarClusterSize){
                maxSimilarClusterSize = similarClusterSize;
                maxIndex = i;
            }
        }
        if(maxSimilarClusterSize == 0)
            return;
        Cluster pivotCluster = clusterList.getElement(maxIndex);
        List similarClusterList = clusterList.getSimilarClusterList(pivotCluster);
        clusterList.mergeClusterList(similarClusterList);
    }
}

```

Figure 8. Feature Similarity based AHC Variant 3: Radius based AHC Algorithm

Let us make some remarks on the three variants of AHC algorithm which are adopted as the approaches to the word clustering. In each iteration of the first variant, cluster pairs with their similarities which are greater than or equal to the threshold are merged. In each iteration of the second variant, a fixed number of cluster pairs with their highest similarities are merged. In each iteration of the third variant, a pivot cluster and its similar clusters are merged. We may derive more variants from the AHC algorithm by setting different policies on merging clusters.

D. System Architecture

This section is concerned with the architecture and the execution flow of the text clustering system. In Section III-C, we describe the graph based AHC variants and adopt them for implementing the text clustering system. In the architecture of the text clustering system, which was previous proposed, the initial AHC algorithm which is mentioned in Section III-B is replaced by its variants. The process of executing the system is to encode the texts into graphs and cluster them. This section is intended to describe the text clustering system which is implemented in this study with respect to its architecture.

A group texts and numerical vectors is illustrated in Figure 9. All of texts are initially given at a time under the assumption of the offline clustering. The texts in the group are encoded into graphs by the process which is described in Section III-A. They are clustered by the AHC algorithm which is described in Section III-C. The online clustering where texts are given as a stream will be considered in next research.

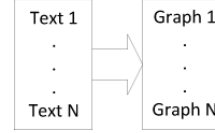


Figure 9. Preparation of Training Examples

The architecture of the text clustering system is illustrated in Figure 10. In the encoding module, texts as clustering targets are encoded into graphs. The clustering module includes the similarity computation module which computes the similarities among the graphs by the process which is described in Section III-A and clusters them by one of the AHC variants. The text clusters are generated as the final output from the system. The difference from the system architecture which was proposed in [8] is to replace the initial AHC algorithm by its variants.

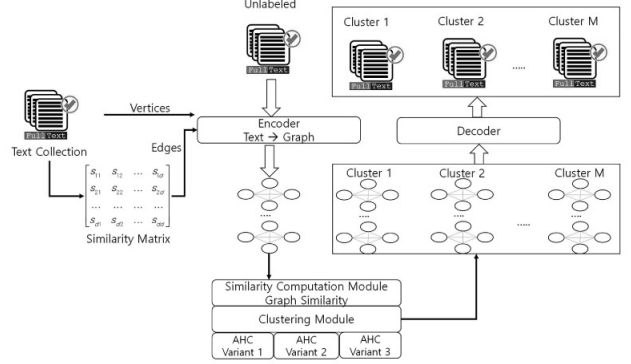


Figure 10. System Architecture

The execution flow of the text clustering system is illustrated in Figure 11. The texts are encoded into graphs. The similarity metric between clusters which is expanded from one between graphs is used in the AHC variants. The data items are clustered, selecting one among the three AHC variants which are described in Section III-C. The external parameter, the similarity threshold, is controlled in the first and the third variant for reaching the desired number of clusters.

Let us make some remarks on the proposed version of the text clustering system whose architecture is presented in Figure 10. The offline clustering is assumed in implementing the system; it will be expanded for doing online clustering

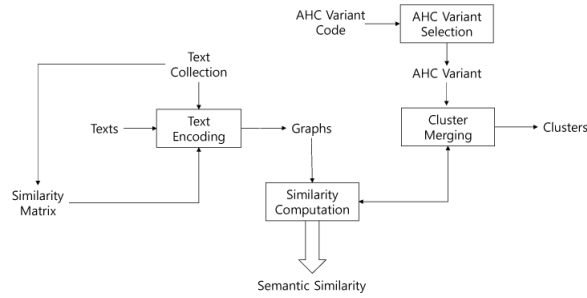


Figure 11. System Flow

in the next research. The upgrading point of the system is to replace the initial AHC algorithm by its variants which are described in Section III-C. The texts are encoded into graphs, the similarity between clusters is computed by equation (6), and they are clustered by one of the AHC variants. It is expected that the clustering performance and speed are improved, by replacing the initial version by its variants.

IV. CONCLUSION

Let us mention the significances of this research as the conclusion. We derive the three variants from the standard version by allowing multiple clusters to be merged at a time. We encode words into graphs and apply the similarity metric among them to the AHC variants as well as the standard version. We design the text clustering system by adopt the AHC variants with the proposed similarity metric. In the next research, we will implement the text clustering system as a real system.

Let us mention the remaining tasks as the further discussions on this research. We will define more operations on graphs and characterize them mathematically. We may modify other machine learning algorithms such as the Naïve Bayes and the SVM (Support Vector Machine) into the graph-based versions. The proposed versions of the AHC variants may be used for other clustering tasks. The text clustering system may be implemented by adopting the proposed AHC variants as real programs.

REFERENCES

- [1] S. Hensman, "Construction of conceptual graph representation of texts", Proceedings of the Student Research Workshop at HLT-NAACL 2004. 2004.
- [2] W. Jin and R. K. Srihari "Graph-based text representation and knowledge discovery", 807-811, The Proceedings of ACM symposium on Applied computing 2007.
- [3] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview", 86-97, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2, 1, 2012.
- [4] K. Sasirekha and P. Baby. "Agglomerative hierarchical clustering algorithm-A Review", 83-85 International Journal of Scientific and Research Publications, 3,1, 2013.

- [5] Z. Nazari, D. Kang, M.R. Asharif, Y. Sung, and S. Ogawa, "A new hierarchical clustering algorithm", 148-152, The Proceedings of IEEE International Conference on Intelligent Informatics and Biomedical Sciences, 2015.
- [6] T. Jo, "Graph based KNN for Optimizing Index of News Articles", 53-62, Journal of Multimedia Information System, 3, 2016.
- [7] T. Jo, "Graph based AHC Algorithm for Text Clustering", 309-314, The Proceedings of Computer Science and Computational Intelligence, 2017.
- [8] T. Jo, "Graph based Version for Clustering Texts in Current Affair Domain", 171-176, The Proceedings of 15st International Conference on Data Science, 2019.
- [9] A.H. Osman and O.M. Barukub, "Graph-based text representation and matching: A review of the state of the art and future challenges", 87562-87583, IEEE Access, Vol 8, 2020.
- [10] T. Jo, "Graph based Version of AHC Algorithm for Document Clustering", in Preparation, 2023.